PROGRAMACIÓN ORIENTADA A OBJETOS

CÓDIGO DE MATERIA 11

Titular: Darío Guillermo Cardacci

2020

FACULTAD DE TECNOLOGÍA INFORMÁTICA

UNIVERSIDAD ABIERTA INTERAMERICANA

Universidad Abierta Interamericana

INDICE

GUÍA DE REVISIÓN CONCEPTUAL.....3
GUÍA DE TRABAJOS PRÁCTICOS 18
GUÍA DE ABORDAJE BIBLIOGRÁFICO.....21

Universidad Abierta Interamericana

GUÍA DE REVISIÓN CONCEPTUAL

UNIDAD I

1. Enumere y explique los aspectos más relevantes que hacen que un software de gran magnitud sea complejo.
2. ¿Cuáles son los cinco atributos de un sistema complejo?
3. ¿Cuáles son las dos jerarquías más importantes que consideramos en la orientación a objetos para sistemas complejos?
4. ¿Con qué podemos enfrentar a la complejidad para obtener partes cada vez más pequeñas y simplificadas del dominio del problema?
5. ¿Cuáles son las dos formas de descomposición más conocidas?
6. ¿Explique en qué se diferencia la descomposición algorítmica y la orientada a objetos?
7. ¿Qué rol cumple la abstracción en la orientación a objetos?
8. ¿Qué rol cumple la jerarquía en la orientación a objetos?
9. ¿Consideraría Ud. al diseño orientado a objetos un desarrollo evolutivo o revolucionario? Justifique.
10. ¿Cuántos y cuáles son los modelos básicos que se manejan en el diseño orientado a objetos?
11. ¿Qué es la programación orientada a objetos?
12. ¿Qué es el diseño orientado a objetos?
13. ¿Qué es el análisis orientado a objetos?
14. ¿Cuáles son los elementos fundamentales en el modelo de objetos?
15. ¿Cuáles son los elementos secundarios del modelo de objetos?
16. Explique el significado de la abstracción.
17. Explique el significado del encapsulamiento.
18. Explique el significado de la modularidad.
19. Explique el significado de la jerarquía.

20. Explique el significado de la tipificación.

21. Explique el significado de la concurrencia.

22. Explique el significado de la persistencia.

23. ¿Cómo se denotan las características esenciales de un objeto que lo distinguen de todos los demás tipos de objetos y proporciona así fronteras conceptuales nítidamente definidas respecto a la perspectiva del observador?

24. ¿A qué denominamos un objeto cliente?

25. ¿A qué denominamos un objeto servidor?

26. ¿A qué denomina Meyer el modelo contractual de programación?

27. ¿Qué establece un contrato entre objetos?

28. ¿Cómo se denomina a las formas en que un objeto puede actuar y/o reaccionar, constituyendo estas formas la visión externa completa, estática y dinámica de la abstracción?

29. ¿Cómo se denomina al conjunto completo de operaciones que puede realizar un cliente sobre un objeto, junto con las formas de invocación u órdenes que admite?

30. ¿A qué nos referimos cuando decimos que un concepto central de la idea de abstracción es el de invariancia?

31. ¿Qué se debe definir para cualquier operación asociada a un objeto?

32. ¿Qué es una precondición?

33. ¿Qué es una postcondición?

34. ¿A qué se denomina excepción?

35. ¿A qué se denomina mensaje?

36. ¿El encapsulado es un concepto complementario a la abstracción? Justifique.

37. ¿Cómo se denomina al elemento de un objeto que captura su vista externa?

38. ¿Cómo se denomina al elemento de un objeto que captura su vista interna la cual incluye los mecanismos que consiguen el comportamiento deseado?

39. ¿El concepto de “ocultar los detalles de implementación” lo asociaría a “esconder los detalles de implementación” o a “evitar el uso inapropiado de los detalles de implementación”? Justifique.

40. ¿Cuáles son los dos aspectos que hacen importante considerar a la modularidad?

41. ¿Para qué se utiliza la jerarquía?

42. ¿Cómo denominamos a la caracterización precisa de propiedades estructurales o de comportamiento que comparten una serie de entidades?

43. ¿Las clases implementan tipos?

44. ¿Los tipos implementan clases?

45. ¿Cómo denominamos a los lenguajes que hacen una comprobación de tipos estricta?

46. ¿Cómo denominamos a los lenguajes que no hacen una comprobación de tipos estricta?

47. ¿A qué se denomina ligadura estática (temprana)?

48. ¿A qué se denomina ligadura dinámica (tardía)?

49. ¿Es lo mismo la comprobación estricta de tipos y la ligadura dinámica?

50. ¿Cómo se denomina la característica que permite a diferentes objetos actuar al mismo tiempo?

51. ¿A qué se denomina concurrencia pesada?

52. ¿A qué se denomina concurrencia ligera o liviana?

53. ¿La concurrencia es la propiedad que distingue un objeto activo de uno que no lo está?

54. ¿Cómo se denomina la característica en orientación a objetos que permite conservar el estado de un objeto en el tiempo y el espacio?

55. ¿Qué cosas se persisten?

56. Defina qué es un objeto desde la perspectiva de la cognición humana.

57. ¿Un objeto es real o abstracto? Justifique.

58. ¿Los objetos poseen límites físicos precisos o imprecisos?

59. ¿Cuáles son las tres cosas que debe tener un objeto?

60. ¿Cuál es la palabra que se puede utilizar como sinónimo de objeto?

61. ¿Cuál es la palabra que se puede utilizar como sinónimo de instancia?

62. ¿Cómo definiría el estado de un objeto?

63. ¿A qué definimos propiedad de un objeto?

64. ¿Qué es lo que distingue a “un objeto” de los “valores simples”?

65. ¿Cómo definiría el comportamiento de un objeto?

66. ¿El comportamiento de un objeto se ve afectado por el estado del mismo o bien que el comportamiento del objeto es función de su estado?

67. ¿Algunos comportamientos pueden alterar el estado de un objeto?

68. Se puede afirmar que el estado de un objeto termina siendo los resultados acumulados de su comportamiento.

69. ¿A qué definiría como operación (método/función miembro)?

70. ¿Cuáles son las tres operaciones más comunes?

71. ¿Cuáles son las dos operaciones habituales que se utilizan para crear y destruir instancias de clases?

72. ¿Qué tipo de operación es el modificador?

73. ¿Qué tipo de operación es el selector?

74. ¿Qué tipo de operación es el iterador?

75. ¿Qué tipo de operación es el constructor?

76. ¿Qué tipo de operación es el destructor?

77. ¿Cómo denominamos operaciones fuera de las clases en aquellos programas orientados a objetos que permiten colocarlas (ej. C++)?

78. ¿Todos los métodos son operaciones?

79. ¿Todas las operaciones son métodos?

80. Dado el protocolo de un objeto (todos sus métodos y subprogramas libres asociados al objeto si el lenguaje lo permite) es conveniente dividirlo en grupos lógicos más pequeños de comportamiento?¿Por qué?

81. ¿Cómo denominamos a los grupos lógicos más pequeños de comportamiento del protocolo total de un objeto?

82. ¿Cuáles son las dos responsabilidades más importantes que posee un objeto?

83. ¿Es relevante el orden en que se invocan las operaciones de un objeto?

84. ¿Por qué decimos que los objetos se pueden considerar como máquinas?

85. ¿Qué es la identidad de un objeto?

86. Dadas dos variable X e Y del mismo tipo ¿qué significa que ambas son iguales?

87. Dadas dos variable X e Y del mismo tipo ¿qué significa asignarle Y a X?

88. Dadas dos variable X e Y del mismo tipo ¿qué significa clonar X en Y?

89. ¿Qué significa realizar una clonación superficial?

90. ¿Qué significa realizar una clonación profunda?

91. ¿Qué es el ciclo de vida de un objeto?

92. ¿Cómo se libera el espacio ocupado por un objeto?

93. ¿Qué tipos de relaciones existen entre los objetos?

94. ¿Cómo podemos definir al enlace entre objetos?

95. ¿Cómo pueden ser los mensajes entre dos objetos en una relación de enlace?

96. ¿Qué es un mensaje unidireccional?

97. ¿Qué es un mensaje bidireccional?

98. ¿Quién inicia el paso de un mensaje entre dos objetos en una relación de enlace?

99. ¿Cuáles son los roles o papeles que puede desempeñar un objeto en una relación de enlace?

100. ¿Qué significa que un objeto actúe como “Actor”?

101. ¿Qué significa que un objeto actúe como “Servidor”?

102. ¿Qué significa que un objeto actúe como “Agente”?

103. Dados dos objetos A y B, si A le puede enviar un mensaje a B, estando ambos relacionados por enlace, decimos que B respecto de A está: ……

104. ¿Cuáles son las cuatro formas de visibilidad que puede poseer un objeto servidor respecto de un objeto cliente?

105. En una relación de enlace de dos objetos, cuando uno le pasa un mensaje al otro, además de adoptar roles ambos deben estar:……..

106. ¿Cuáles son las posibles formas de sincronización?
107. ¿Qué significa que dados dos objetos A y B estos están secuencialmente sincronizados?
108. ¿Qué significa que la forma de sincronizarse de un conjunto de objetos es vigilada?
109. ¿Qué significa que la forma de sincronizarse de un conjunto de objetos es síncrona?
110. ¿El enlace es una relación de igual a igual o jerárquica?
111. ¿La agregación es una relación de igual a igual o jerárquica?
112. ¿Qué tipo de jerarquía denota la agregación?
113. ¿Qué otros nombres recibe el “todo” en una relación de agregación?
114. ¿En una relación de agregación las “partes” forman parte del estado del “todo”?
115. ¿Qué tipos de agregación existen?
116. ¿Qué caracteriza a la agregación con contención física?
117. ¿Qué es una clase?
118. ¿La interfaz de la clase proporciona su visión interna?
119. ¿La implementación de la clase proporciona su visión externa?
120. ¿En cuántas partes la podemos dividir una interfaz en términos de la accesibilidad o visibilidad que posee?
121. ¿Qué tipos básicos de relaciones existen entre las clases?
122. ¿Qué relaciones entre clases se desprenden de las tres relaciones básicas?
123. ¿La asociación denota una dependencia semántica y la dirección de esta asociación?
124. ¿Qué significa la cardinalidad en una relación?
125. ¿Qué cardinalidad puede existir entre clases relacionadas por asociación?
126. ¿Qué es la herencia?
127. ¿Cuántos tipos de herencia existen?
128. ¿A qué se denomina herencia simple?
129. ¿A qué se denomina herencia múltiple?

130. ¿Cómo se denomina a la clase que no se espera tener instancias de ella y solo se utilizará para heredar a otras clases?

131. ¿Cómo se denomina a la clase que se espera tener instancias de ella y puede utilizarse para heredar a otras clases o no?

132. ¿Cómo se denomina al método de una clase abstracta que no posee implementación y fue pensado para que sea implementado en las sub clases que lo heredan?

133. ¿Cómo se denomina a la clase más generalizada en una estructura de clases?

134. ¿Qué es el polimorfismo?

135. ¿Cómo se denomina cuando una clase posee métodos que comparten el nombre y se diferencias por su firma?

136. ¿Qué sentencias de código se evitan utilizar cuando se aplica correctamente el polimorfismo?

137. ¿Qué es la agregación cómo relación entre clases?

138. ¿Qué formas de contención física existen en la agregación?

139. ¿Qué características posee la contención física por valor?

140. ¿Qué características posee la contención física por referencia?

141. ¿Qué es una relación de uso?

142. ¿Qué es la instanciación?

143. ¿Todo objeto es una instancia de una clase?

144. ¿Qué es una metaclase?

145. ¿Qué métricas hay que observar para determinar la calidad de una abstracción?

146. ¿Qué es el acoplamiento?

147. ¿Qué es la cohesión?

148. ¿Qué es la suficiencia?

149. ¿Qué es la compleción?

150. ¿Qué significa ser primitivo?

151. ¿Qué se debe observar al momento de decidir si una abstracción debe implementar un determinado comportamiento o no?

152. ¿Qué formas puede adoptar el paso de un mensaje?

153. ¿Qué características posee un mensaje síncrono?

154. ¿Qué características posee un mensaje de abandono inmediato?

155. ¿Qué características posee un mensaje de intervalo?

156. ¿Qué características posee un mensaje Asíncrono?

157. ¿Qué significa que una abstracción está accesible a otra?

158. ¿Qué expresa la ley de Demeter?

159. ¿Cuál es la consecuencia inmediata de aplicar la ley de Demeter?

160. ¿Cuáles son las cuatro formas fundamentales por las cuales un objeto X puede hacerse visible a un objeto Y?

161. ¿Para qué sirve clasificar a los objetos?

162. ¿Por qué es tan difícil la clasificación de objetos?

163. ¿Cómo es el rol del observador en la clasificación de objetos?

164. ¿Cuáles son las aproximaciones generales a la clasificación?

165. ¿Qué es la categorización clásica?

166. ¿Qué es el agrupamiento conceptual?

167. ¿Qué es la teoría de prototipos?

168. ¿Qué es una abstracción clave?

169. ¿Qué son los mecanismos?

UNIDAD II

170. ¿Qué es un campo de una clase?

171. ¿Qué es un método de una clase?

172. ¿A qué denominamos sobrecarga?

173. ¿Qué es una propiedad de una clase?

174. ¿Qué tipos de propiedades existen?

175. ¿Qué ámbitos pueden tener los campos, métodos y propiedades de las clases?

176. ¿Qué características posee cada ámbito existente si se lo aplico a un campo, un método y una propiedad de una clase?

177. ¿Para qué se utilizan los constructores?

178. ¿A qué se denomina tiempo de vida de un objeto?

179. ¿Para administrar las instancias de .NET se utiliza un contador de referencias?

180. ¿Qué objeto es el encargado de liberar el espacio ocupado por objetos que ya no se utilizan?

181. ¿Qué son los sucesos?

182. ¿Qué se utiliza para declarar un suceso?

183. ¿Cómo se logra que ocurra un suceso?

184. ¿Cómo se pueden atrapar los sucesos?

185. ¿Cómo le indica a un evento que desea cambiar el tipo del argumento que por defeto es EventArgs?

186. ¿Cómo y qué cosas se pueden compartir en una clase?

187. ¿Qué características poseen los campos compartidos?

188. ¿Qué características poseen los métodos compartidos?

189. ¿Qué características poseen los sucesos compartidos?

190. ¿Qué son y para que se pueden utilizar las clases anidadas?

191. ¿Qué ámbitos / modificadores de acceso existen? Explique las caracteríStcas de cada uno.

192. ¿Qué cosas se pueden heredar?

193. ¿Cómo y para qué se puede aprovechar en la práctica el polimorfismo?

194. ¿Cómo y para qué se utiliza la clase derived?

195. ¿Qué representa this?
196. ¿Qué clase representa a la clase base?
197. ¿Para qué se usa una clase abstracta?
198. ¿Para qué se usa una clase sellada?
199. ¿Qué es la sobreescritura?
200. ¿Qué elementos se pueden sobreescribir?
201. ¿A qué se denomina sombreado de métodos?
202. ¿Qué característica peculiar posee el sombrado vs la sobre-escritura?

UNIDAD III

203. ¿Qué es un Framework?
204. ¿Qué son los frozen-spots en un Framework?
205. ¿Qué son los hot-spots en un Framework?
206. ¿Cómo se puede clasificar un Framework según su extensibilidad?
207. ¿Qué es un Framework de Caja Blanca?
208. ¿Qué es un Framework de Caja Negra?
209. ¿Qué ventajas posee utilizar un Framework?
210. ¿Qué problemas resuelve .NET Framework?
211. ¿Qué es y qué permite hacer el CLR?
212. ¿Qué es el MSIL?
213. ¿Qué es el CTS?
214. ¿Qué es el CLS?
215. ¿Qué es una excepción?
216. ¿Qué se coloca en el bloque “Catch”?

217. ¿Cómo construiría un objeto del tipo “Exception” personalizado?

218. ¿Qué ocurre si en el bloque de código donde se produce la excepción el error no está siendo tratado?

219. ¿Cuál es el objeto de mayor jerarquía para el manejo de excepciones?

220. ¿En qué namespace se encuentra la clase Exception?

221. ¿Cuáles son las dos clases genéricas más importantes definidas en el FrameWork además de Exception?

222. ¿Qué instrucción se utiliza para poner en práctica el control e interceptar las excepciones?

223. ¿Dónde se coloca el código protegido contra excepciones si se iniciara una excepción?

224. ¿Qué tipo de excepción se utiliza para interceptar un error de división por cero?

225. ¿Qué tipo de excepción se utiliza para interceptar una DLL que tiene problemas al ser cargada?

226. ¿Qué colocaría dentro de una clausula “Catch” para especificar una condición adicional que el bloque “Catch” deberá evaluar como verdadero para que sea seleccionada?

227. ¿Si se desea colocar código de limpieza y liberación de recursos para que se ejecute cuando una excepción se produzca, dónde lo colocaría?

228. ¿Qué instrucción se utiliza para provocar un error y que el mismo se adapte al mecanismo de control de excepciones?

229. ¿Escriba el código que permitiría provocar una excepción del tipo “ArgumentException”?

230. ¿Cómo construiría un objeto del tipo “Exception” personalizado?

231. ¿Cómo armaría un “Catch” personalizado para que se ejecute cuando se de la excepción “ClienteNoExisteException”?

232. ¿Dónde se encuentran las instancias de los objetos administrados por el GC?

233. ¿Cuáles son los dos métodos más notorios que deben implementar las clases para trabajar correctamente con la recolección de elementos no utilizados y matar las instancias administradas y no administradas?

234. ¿Qué método se utiliza para que el GC recolecte los elementos no utilizados?

235. ¿Qué método se utiliza para suspender el subproceso actual hasta que el subproceso que se está procesando en la cola de finalizadores vacíe dicha cola?

236. ¿Qué método se ejecuta en los objetos cuando se ejecuta el método collect del GC?

237. ¿Qué método debería exponer una clase bien diseñada teniendo en consideración que no posee destructor?

238. ¿Cómo obtengo el método “Dispose”?

239. ¿Qué se programa en el método “Dispose”?

240. ¿Se pueden combinar el uso de “Dispose” y el Destructor?

241. ¿A qué se denomina “Resurrección de Objetos”?

242. ¿A qué se denomina “Generación” en el contexto de la recolección de elementos no utilizados?

243. ¿Qué valores puede adoptar la “Generación” de un objeto?

244. ¿Cómo se puede obtener el número de veces que se ha producido la recolección de elementos no utilizados para la generación de objetos especificada?

245. ¿Cómo se obtiene el número de generación actual de un objeto?

246. ¿Cómo se puede recuperar el número de bytes que se considera que están asignados en la actualidad?

247. ¿Qué utiliza para convertir un objeto en “no” válido para la recolección de elementos no utilizados desde el principio de la rutina actual hasta el momento en que se llamó a este método?

248. ¿Cómo se solicita que el sistema no llame al finalizador del objeto especificado?

249. ¿Cómo se solicita que el sistema llame al finalizador del objeto especificado, para el que previamente se ha llamado a “SuppressFinalize”?

250. ¿Cómo se obtiene el número máximo de generaciones que el sistema admite en la actualidad?

UNIDAD IV

251. ¿Para qué se utilizan las interfaces?

252. ¿Cómo se implementa una interfaz?

253. ¿Se pueden heredar las interfaces?

254. ¿Se puede implementar un tipo de polimorfismo peculiar por medio de interfaces?
255. ¿Para qué se utiliza la interfaz IComparable?
256. ¿Para qué se utiliza la interfaz IComparer?
257. ¿Para qué se utiliza la interfaz ICloneable?
258. ¿Para qué se utiliza la interfaz IEnumerable?
259. ¿Para qué se utiliza la interfaz IEnumerator?
260. ¿Qué es un delegado?
261. ¿A qué elementos se les puede delegar?
262. ¿Qué se puede delegar?
263. ¿Cómo construiría un delegado?
264. ¿Cómo implementaría un procedimiento de devolución de llamadas?
265. ¿Para qué sirve la multidifusión de delegados?

UNIDAD V

266. ¿Qué son y para qué se usan los tipos genéricos?
267. ¿A partir de que versión de c# se pueden utilizar?
268. Enumere las ventajas de utilizar genéricos.
269. ¿A qué elementos se le pueden aplicar genericos?
270. ¿Cuáles son los usos más comunes de los genéricos?
271. ¿Qué aspectos trascendentes se deben considerar al crear clases genéricas?
272. ¿Cués es la diferencia entre heredar de una clase genérica abierta y una clase genérica cerrada?
273. ¿Por qué es importante colocar restricciones en los parámetros de tipo?
274. ¿Cuáles son los tipos de restricciones que se le pueden colocar a un parámetro de tipo?
275. ¿Ejemplifique cómo crearía un método genérico con un parámetro de tipo?

276. ¿Qué es LinQ?
277. ¿Qué orígenes de datos se pueden consultar con LinQ?
278. ¿Cuáles son las partes básicas de una consulta LinQ?
279. ¿Qué especifica la consulta en una estructura de LinQ?
280. Mencione al menos tres clausulas (las más importantes) que se usan en una expresión de consulta LinQ.
281. Explique que hace cada cláusula enumerada en la pregunta anterior
282. Dado que una expreseión de consulta genera un Ienumerable, enumere y explique los métodos de extensión que posee Ienumerable. (p.e Count)
283. ¿Qué utilizaría para ordenar en una expresión LinQ?
284. ¿Qué utilizaría para lograr una unión entre dos origenes de datos en una expresión LinQ?
285. ¿Cómo se pueden generar nuevos tipos utilizando LinQ?
286. ¿Qué es una expresión Lambda?
287. Ejemplifique cómo se puede utilizar una expresión lambda para realizar una consulta.
288. ¿Qué debo realizar para crear una expresión lambda?
289. ¿Indique como puede declarar un delegado utilizndo Func y que significa cada elemento utilizado?
290. Crear un delegado utilizando Func que posea tres parámetros de entrada y uno de salida. Los parámetros de entrada son: el primero int, el segundo double y el tercero bool. El parámetro de retorno es de tipo bool.

UNIDAD VI

291. ¿Qué características posee un esquema cliente - servidor?
292. ¿Qué significa pasar información batch?
293. ¿Qué significa pasar información on line?
294. ¿Qué es un protocolo?

295. ¿Qué protocolo usa una red de área local?

296. ¿Qué protocolo usa internet?

297. ¿Qué hace el protocolo IP?

298. ¿Qué ventajas tiene distribuir procesos?

299. ¿Qué ventajas tiene distribuir almacenamientos?

300. ¿Qué es un socket?

301. ¿Qué característica posee un socket sincrónico?

302. ¿Qué característica posee un socket asincrónico?

303. ¿Qué objeto se puede utilizar para construir un navegador?

304. ¿Para qué se utilizan los puertos de la pc?

305. ¿Qué puertos posee una PC?

306. ¿Cómo funciona el puerto paralelo?

307. ¿Cómo funciona el puerto serie?

308. ¿Cómo funciona el puerto USB?

309. ¿Qué es la domótica?

GUÍA DE TRABAJOS PRÁCTICOS

UNIDAD I

1. Desarrollar una jerarquía de clases relacionadas por herencia y agregación que represente la estructura necesaria para identificar claramente los elementos encontrados en un sistema de calificaciones de una universidad. En cada clase detallar métodos y propiedades.

2. Defina una situación donde se desee llegar desde un estado inicia a uno final, por ejemplo, una cuenta bancaria que posee un saldo inicial y se la somete a una operación bancaria. Desarrollar la clase cuenta con las propiedades y métodos que considere pertinentes y demuestre como cambia el estado del objeto en la medida que se le realizan depósitos, extracciones y transferencias.

3. Genere una estructura de clases donde se pueda observar la herencia simple y se justifique utilizar el polimorfismo.

4. Genere una estructura de clases donde se pueda observar la herencia múltiple y se justifique utilizar el polimorfismo.

5. Desarrollar conjuntos de objetos reales donde para su agrupación y clasificación sea necesario aplicar la categorización conceptual, la clásica y el agrupamiento prototípico. Justifique.

UNIDAD II

1. Desarrollar un programa que posea una estructura de clases donde se puedan observar dos métodos y el constructor sobrecargados.

2. Desarrollar un programa que posea una estructura de clases donde se pueda observar una propiedad de solo lectura, una propiedad de solo escritura, una propiedad de escritura-lectura, una propiedad de predeterminada y una propiedad con argumentos.

3. Desarrollar un programa que posea una clase que aplique un destructor que finalice el ciclo de vida de un onjeto que se instranció en el constructor.

4. Desarrollar un programa que posea una clase que tenga propiedades, métodos y sucesos. Los sucesos deben tener suscripciones manuales realizadas por el programador a funciones de la clase donde se en cuentra instanciado el objeto que posee los sucesos.

5. Desarrollar un programa que posea una clase que contenga al menos dos campos, tres métodos, un constructor y dos sucesos estáticos. Comente que característica le otorga esta característica a cada miembro.

6. Desarrollar un programa donde se observe claramente el uso de los miembros de base en un entorno de herencia. Enfatice las particularidades al usarlo en los constructores y

finalizadores. Demuestre en el mismo programa el uso de this. Establezca las diferencias y en qué caso se justifica utilizar cada uno.

7. Desarrollar un programa que posea al menos una clase abstracta, un método virtual, una clase sellada y una clase anidada.

UNIDAD III

8. Desarrollar un programa que genera varias instancias (una cantidad importante), verifique la memoria utilizada, pase el GC y vuelva a verificar el espacio de memoria. ¿Qué se observa?

9. Desarrollar un programa que genere una instancia, pierda la referencia a la misma y aplicando la técnica de "resurrección de objetos" logre obtener la referencia a ese mismo objeto.

10. Desarrollar un programa que aplique el concepto de manejo de errores. La estructura propuesta debe tener al menos 5 Catch y el finally.

11. Desarrollar un programa que aplique el concepto de manejo de errores. Generar un error personalizado por medio de una clase que herede de Exception y disparar el error con Throw.

UNIDAD IV

12. Desarrollar un programa que implemente la interfaz IComparable. Genere un arrary de objetos y ordénelos.

13. Desarrollar un programa que implemente la interfaz IComparer. La clase que lo implementa deberá tener al menos cinco criterios de ordenamiento. Genere un arrary de objetos y ordénelos por cada criterio.

14. Desarrollar un programa que implemente la interfaz ICloneable. Clone el objeto que posee la interfaz y demuestre que la clonación funcionó.

15. Desarrollar un programa que implemente las interfaces IEnumerable e IEnumerator. Lo adaptado recórralo con el for each y muestre los resultados.

UNIDAD V

1. Desarrollar un programa que aplique el concepto genéricos a nivel de clase y en almenos dos métodos.

2. Desarrollar un programa que utilice Linq para realizar consultas.

3. Desrrollar un programa que utilice expresiones lambda.

UNIDAD VI

4. Desarrollar un programa que utilizando socket permita escribir en una aplicación que se esté ejecutando en una PC de la red y lo escrito se pueda visualizar en la aplicación que está corriendo en otra PC de la red.

5. Desarrollar un programa que utilizando socket que emule un servidor de chat. También desarrollar el software cliente que se ejecutará en las Pc's que se conecten al servidor. Cada mensaje enviado por un cliente lo podrán recibir todos los demás. En el caso que un cliente solicite comunicación privada el mensaje se verá solo por los que participan de esta comunicación. Un cliente puede solicitar armar un grupo y el resto unirse a él, para que esto suceda el propietario del grupo deberá aceptarlos y tiene la potestad de desconectarlos. El servidor es quien modera todo y por allí pasan todos los mensajes.

6. Desarrollar un programa que haciendo uso del puerto paralelo y un emulador que permita enviar señales al mismo y recolectar las entradas que se producen en él.

GUÍA DE ABORDAJE BIBLIOGRÁFICO

UNIDAD I

Cardacci Dario y Booch, Grady. **Orientación a Objetos. Teoría y Práctica.** -- Buenos Aires, Argentina. Pearson Argentina, 2013. Capítulos 1 al 4.

Deitel Harvey M. Y Paul J. Deitel. **Cómo programar en C#**. Segunda edición. Pearson. México 2007. Capítulos 9 al 11

UNIDAD II

Cardacci Dario y Booch, Grady. **Orientación a Objetos. Teoría y Práctica.** -- Buenos Aires, Argentina. Pearson Argentina, 2013. Capítulos 5.1 al 5.6

Deitel Harvey M. Y Paul J. Deitel. **Cómo programar en C#**. Segunda edición. Pearson. México 2007. Capítulos 9 al 11

UNIDAD III

https://docs.microsoft.com/es-es/dotnet/csharp/language-reference/keywords/exception-handling-statements

Balena, Francesco. **Programación avanzada con Visual Basic 2005.** -- México, DF: McGraw-Hill Interamericana, c2008. Capítulo 1.

Deitel Harvey M. Y Paul J. Deitel. **Cómo programar en C#**. Segunda edición. Pearson. México 2007. Capítulos 12.

UNIDAD IV

https://docs.microsoft.com/es-es/dotnet/csharp/tour-of-csharp/interfaces

https://docs.microsoft.com/es-es/dotnet/csharp/tour-of-csharp/delegates

UNIDAD V

Deitel Harvey M. Y Paul J. Deitel. **Cómo programar en C#**. Segunda edición. Pearson. México 2007. Capítulos 25

https://docs.microsoft.com/es-es/dotnet/csharp/programming-guide/concepts/linq/

https://docs.microsoft.com/es-es/dotnet/csharp/programming-guide/generics/

https://docs.microsoft.com/es-es/dotnet/csharp/programming-guide/concepts/linq/linq-and-generic-types

https://docs.microsoft.com/es-es/dotnet/csharp/programming-guide/statements-expressions-operators/lambda-expressions

UNIDAD VI

MSDN. Microsoft Developer Network.
http://msdn.microsoft.com/en-us/library/system.net.sockets.socket.aspx

Cardacci Dario y Booch, Grady. **Orientación a Objetos. Teoría y Práctica.** -- Buenos Aires, Argentina. Pearson Argentina, 2013. Capítulos 5.7.

Deitel Harvey M. Y Paul J. Deitel. **Cómo programar en C#**. Segunda edición. Pearson. México 2007. Capítulos 23

Montefinal, Fabián H.; Cardacci, Darío G. **Conceptos básicos sobre electricidad: electrónica y puertos de la PC**. 2a.ed.-- Buenos Aires: Universidad Abierta Interamericana, c2006. 78 páginas